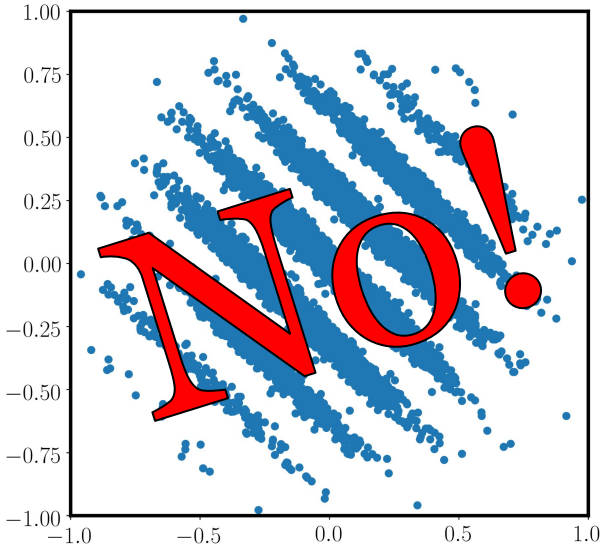
Continuous Learning with Errors

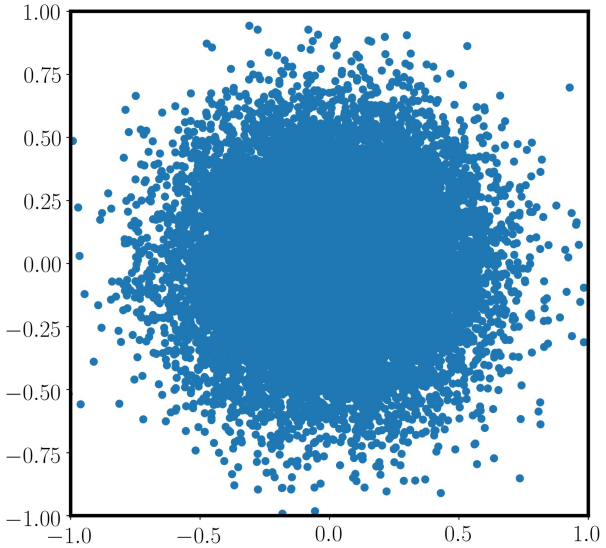
Min Jae Song (NYU) and Yi Tang (UMich)
Joint work with Joan Bruna and Oded Regev (NYU)
June 8th 2021, STOC

Motivation: Gaussian pancakes

Is this the standard Gaussian distribution?



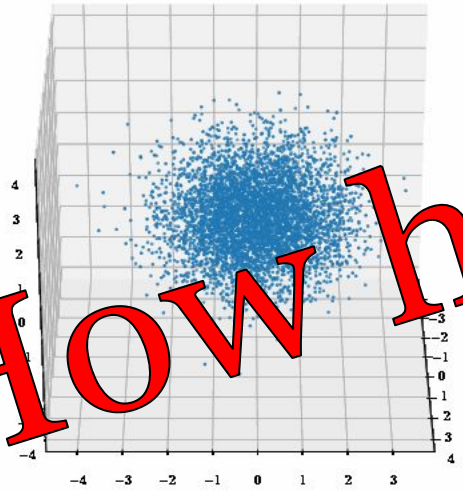
Gaussian pancakes



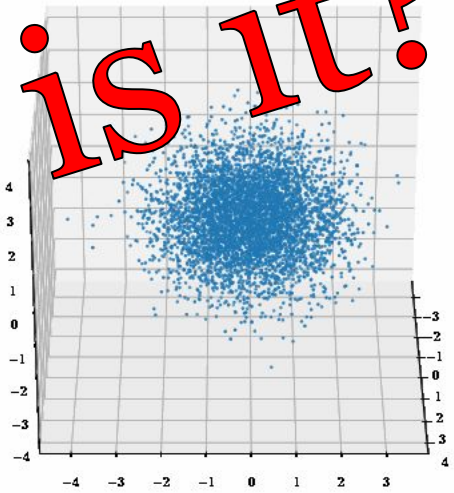
Standard Gaussian

Motivation: Gaussian pancakes

But what if you hide this discrete direction in higher dimensions?



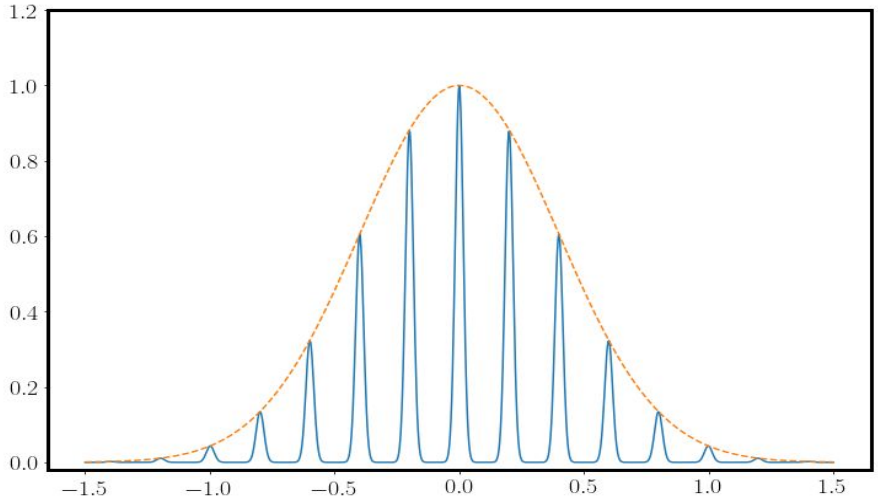
Gaussian pancake!



Standard Gaussian

How hard is it?

Gaussian pancakes



The Gaussian pancakes distribution is a noisy discrete Gaussian (blue) in one hidden direction. In other n-1 directions, the distribution is Gaussian (orange).

For reasons we will explain later, we call this distribution the **homogeneous Continuous Learning with Errors (hCLWE)** distribution.

SQ-hardness of distinguishing Gaussian pancakes [DiakonikolasKaneStewart17]

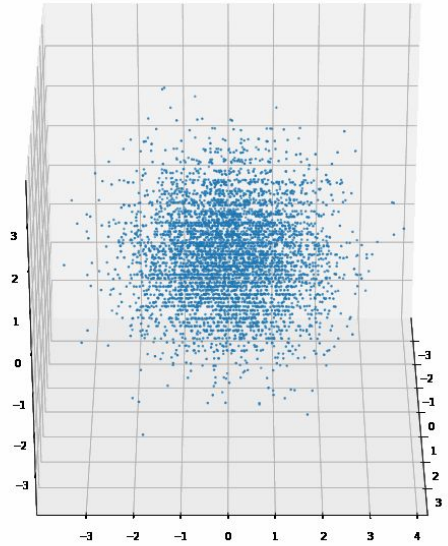
Thm [DKS17]: Distinguishing Gaussian pancakes from the standard Gaussian is hard for **statistical query (SQ) algorithms**.

Def: an **SQ algorithm** [Kearns98] accesses the input distribution only indirectly from noisy expectations. It can query the distribution with any bounded function f: $\mathbb{R}^n \to [-1,1]$, and receive a noisy version of $\mathbb{E}[f(x)]$, instead of getting individual samples.

- can be seen as an abstraction capturing a wide range of algorithms that operate only with distributional quantities rather than per-sample quantities, e.g., full-batch gradient descent.

SQ-hardness of distinguishing Gaussian baguettes [BubeckLeePriceRazenshteyn19]

Thm [BLPR19]: Similarly, still SQ-hard when you have **multiple** discrete directions (Gaussian “baguettes”).



Implications of hardness of distinguishing Gaussian pancakes

1. **Improperly learning** (= density estimation) mixtures of Gaussians is hard for SQ algorithms, even when the components are nearly non-overlapping and **parameter recovery** is info-theoretically possible with poly(n) samples [DKS17].
2. Learning **robust*** classifiers can be hard, even when they exist and are learnable info-theoretically [BLPR19].

* Robust in the sense that the classifier is not vulnerable to small, imperceptible input perturbations.

Open question by [BLPR19]:

Is detecting the pancake structure computationally hard for **any** algorithm?

We resolve this here in the **affirmative**.

Adversarial examples: why learning robust classifiers is important

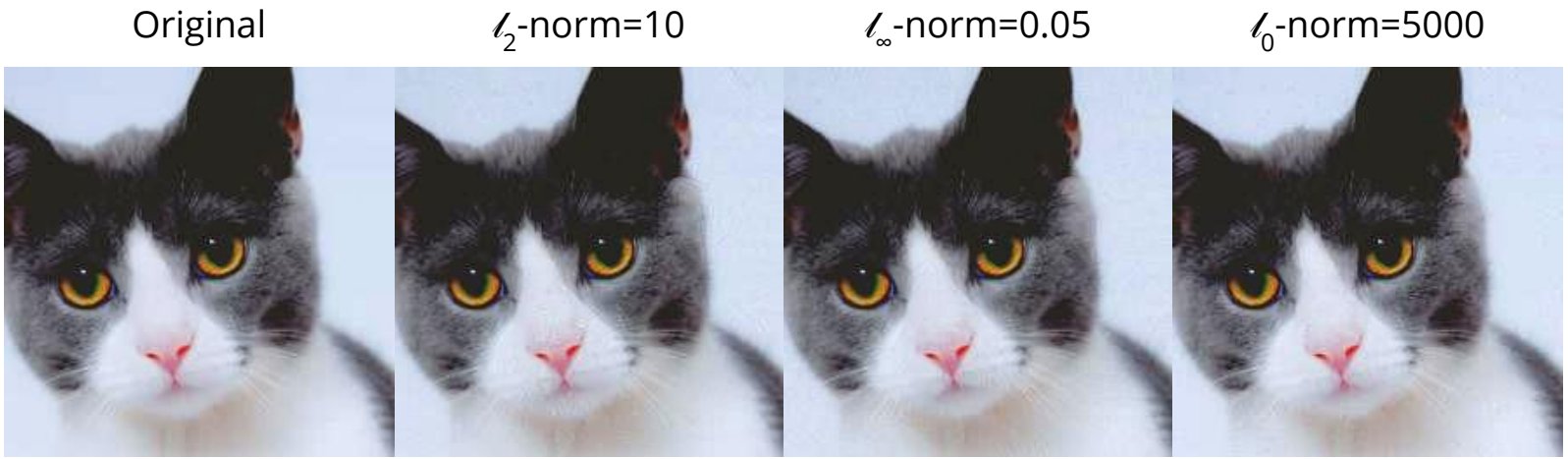
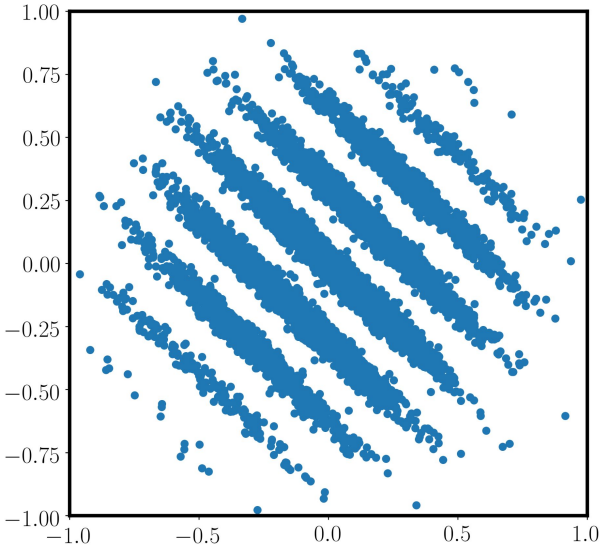


Figure from [ShafahiHuangStuderFeiziGoldstein20]

[BLPR19] suggests that even though robust classification might be *possible* information-theoretically, *learning* a robust classifier might be computationally intractable (in theory).

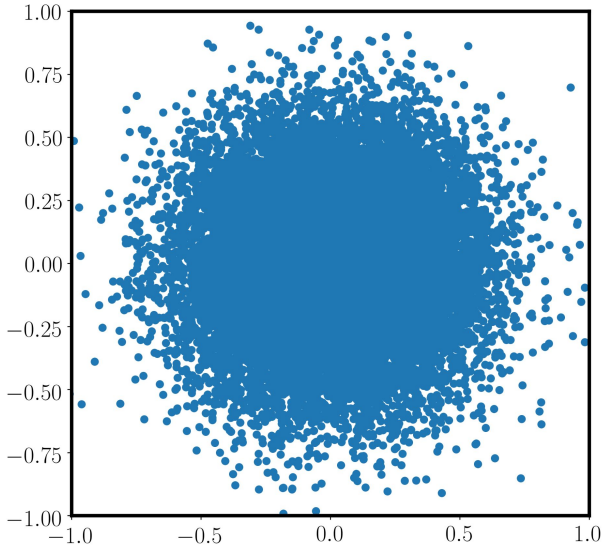
Our goal: distinguishing hCLWE

Prove that distinguishing the following two distributions (in high dimension) is computationally hard.



Gaussian pancakes

vs

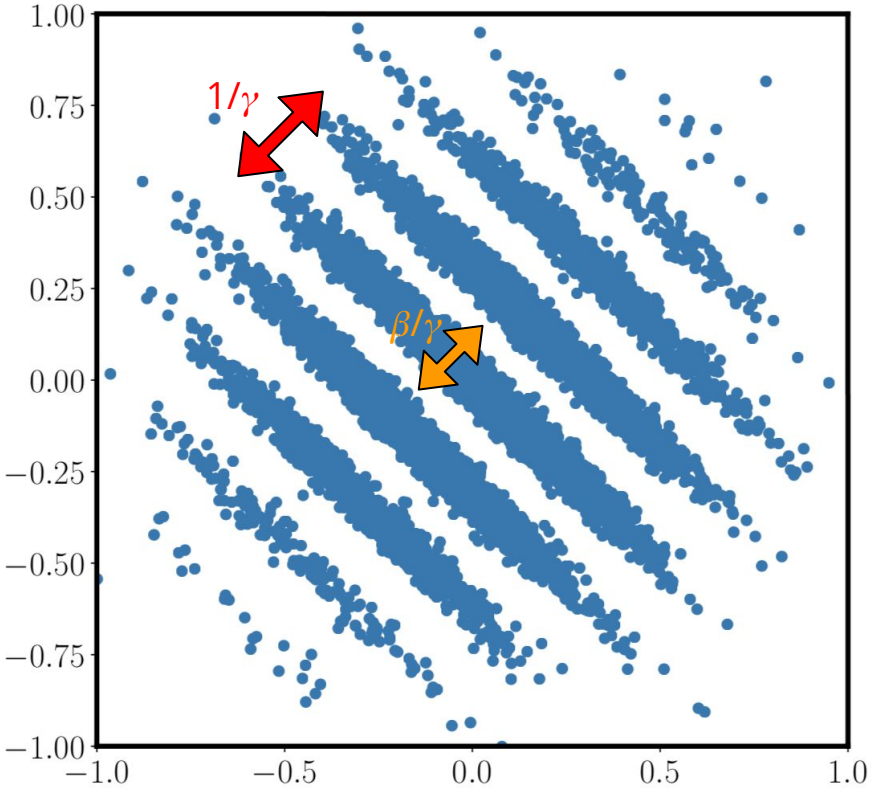


Standard Gaussian

hCLWE (aka Gaussian pancakes)

γ : pancake (inverse) spacing

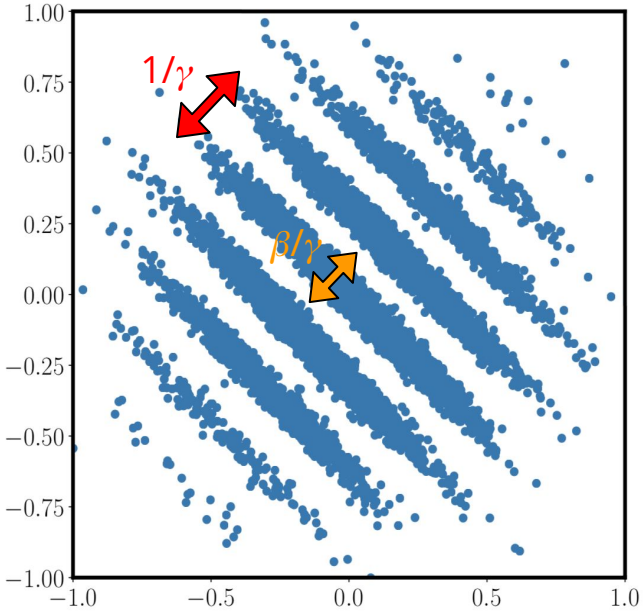
β : pancake (relative) thickness



Our result: hardness of Gaussian pancakes

Distinguishing Gaussian pancakes with spacing $1/\gamma$ less than $n^{-1/2}$ from the standard Gaussian with accuracy **slightly (inverse-polynomially) better than chance** is computationally hard, unless there are polynomial-time **quantum** algorithms for certain fundamental **worst-case** lattice problems.

* β can be any inverse polynomial less than 1.



Implications of our hardness result

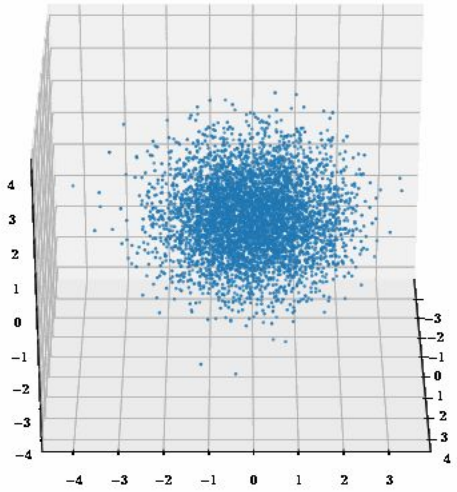
Assuming some worst-case lattice problems cannot be solved by polynomial-time quantum algorithms …

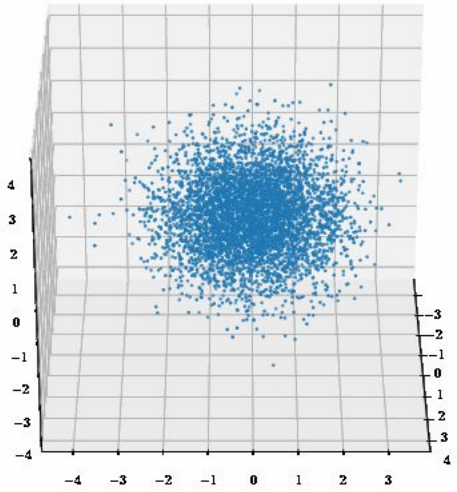
- Distinguishing Gaussian pancakes/baguettes from the standard Gaussian is hard for **any** polynomial-time algorithm. In fact, hard for baguettes with $O(n)$ many discrete directions.
- Improperly learning mixtures of Gaussians is computationally hard even when the mixture components are nearly non-overlapping.

Hardness of improper learning is generally difficult to show because there is no restriction on what hypothesis the learning algorithm can output.

Worst-case to average-case reduction: these are **average-case** hardness results based on **worst-case** hardness assumptions. Only a few hardness of improper learning results are based on worst-case hardness. [KlivansSherstov06].

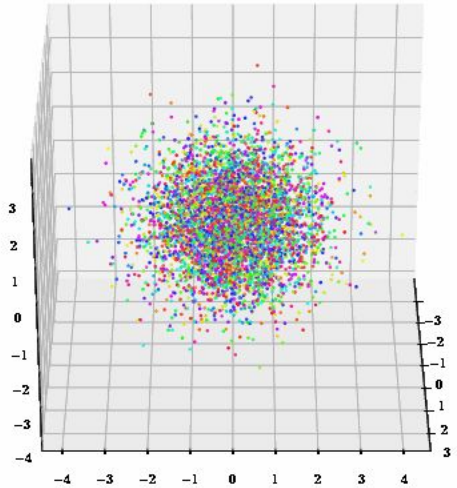
Hardness of (h)CLWE: proof overview

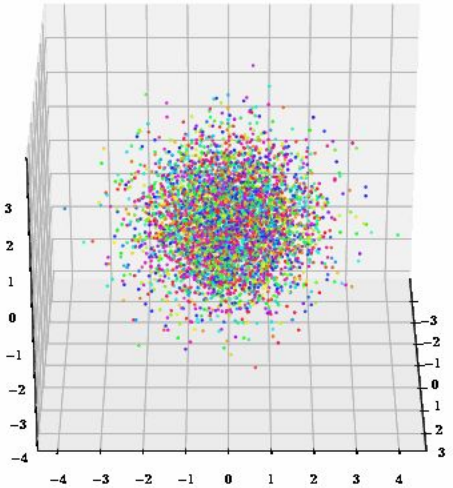




Hardness of (h)CLWE: proof overview

We prove a stronger hardness result, for a relaxed problem: (*inhomogeneous*) CLWE.





Hardness of (h)CLWE: proof overview

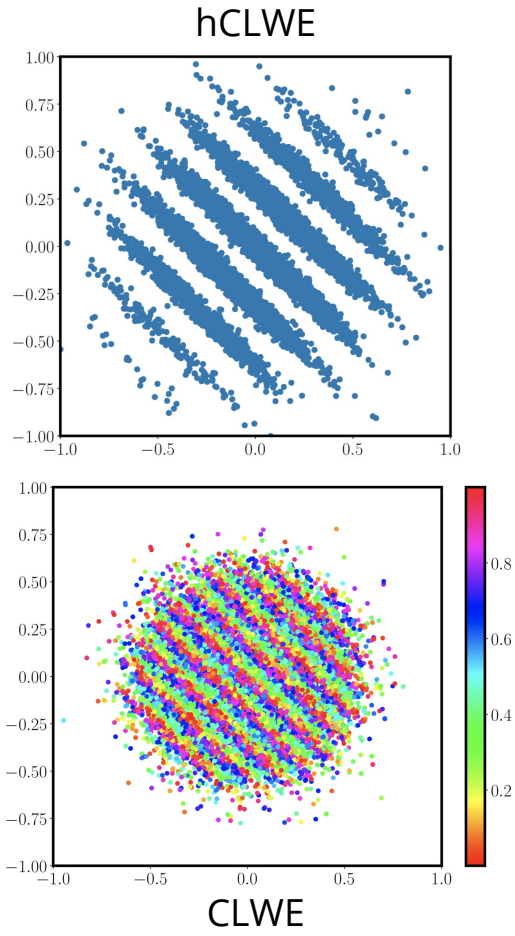
Def. (β,γ)-CLWE: To decide whether the given samples of the form $(\mathbf{y},z)$ with $\mathbf{y} \sim \mathcal{N}(0,I_n)$ have either:

(1) periodic "colors" z along some secret direction $\mathbf{w} \in \mathbb{R}^n$, i.e., $z = (\gamma\langle\mathbf{y},\mathbf{w}\rangle + e) \bmod 1$ where $e \sim \mathcal{N}(0,\beta)$, or

(2) uniformly random "colors" $z \in [0,1)$.

hCLWE samples are roughly CLWE samples with $z = 0$.

We show the hardness results by reducing *worst-case* lattice problems to CLWE, and reducing CLWE to hCLWE via rejection sampling by $z \approx 0$.

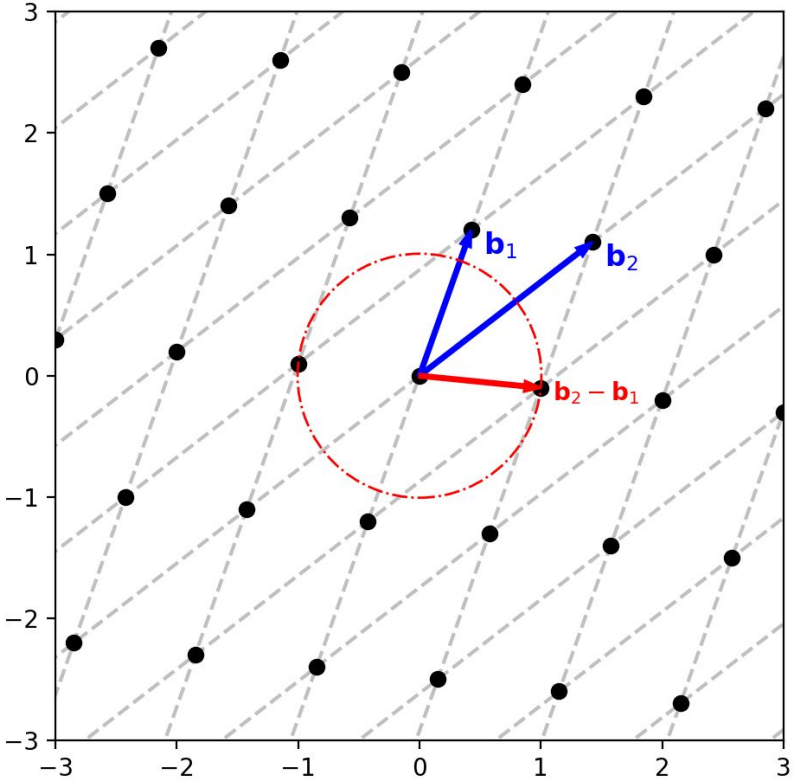


Lattices and lattice problems

For a basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ of $\mathbb{R}^n$, the lattice L generated by the basis is the set of all *integer* linear combinations of the basis vectors.

The minimum distance $\lambda_1(L)$ is the shortest length of nonzero lattice vectors in lattice L.

The Shortest Vector Problem (SVP): To find a lattice vector with the shortest length $\lambda_1(L)$ for a given lattice L.



Hardness based on SVP

Def. Promise version of SVP (φ-GapSVP): Given a lattice L, the goal is to decide whether $\lambda_1(L) \le 1$ or $\lambda_1(L) > \varphi$.

φ-GapSVP is believed to be hard (even *quantumly*) for any polynomial $\varphi=\varphi(n)$.

- Fastest known algorithms (for some small poly φ) run in time $2^{O(n)}$.
- NP-hard for any constant φ. [Micciancio01,Khot05]

[Regev05,PeikertRegevStephens-Davidowitz17] show a (*quantum*) reduction from $O(n/\alpha)$-GapSVP to (α,q)-LWE (for large enough $\alpha \cdot q$).
We follow the same framework and reduce $O(n/\beta)$-GapSVP to (β,γ)-CLWE, for poly $\gamma \ge 2n^{1/2}$ and inverse-poly β.

Learning with Errors (LWE)

Def. (α,q)-LWE: To decide whether the given samples of the form $(\mathbf{a},b)$ with $\mathbf{a} \sim (\mathbb{Z}/q\mathbb{Z})^n$ have either:

(1) periodic b along some secret direction $\mathbf{s} \in (\mathbb{Z}/q\mathbb{Z})^n$, i.e., $b = (\langle \mathbf{a},\mathbf{s} \rangle/q + e) \bmod 1$ where $e \sim \mathcal{N}(0,\alpha)$, or

(2) uniformly random $b \in [0,1)$.

Remark: By discretizing with $b' = \lfloor q \cdot b \rfloor \in \mathbb{Z}/q\mathbb{Z}$, the search version (to find secret $\mathbf{s}$ given periodic b) can be viewed as solving system of linear equations *with errors* over $\mathbb{Z}/q\mathbb{Z}$, of the form $\langle \mathbf{a},\mathbf{s} \rangle \approx b'$.

Analogies between CLWE and LWE

(β,γ)-CLWE	(α,q)-LWE
secret $\mathbf{w} \in \mathbb{R}^n$, $\|\mathbf{w}\| = 1$	secret $\mathbf{s} \in (\mathbb{Z}/q\mathbb{Z})^n$
samples $(\mathbf{y},z)$	samples $(\mathbf{a},b)$
$\mathbf{y} \sim \mathcal{N}(0,I_n)$	$\mathbf{a} \sim (\mathbb{Z}/q\mathbb{Z})^n$
$z = (\gamma\langle\mathbf{y},\mathbf{w}\rangle + e) \bmod 1$ where $e \sim \mathcal{N}(0,\beta)$	$b = (\langle\mathbf{a},\mathbf{s}\rangle/q + e) \bmod 1$ where $e \sim \mathcal{N}(0,\alpha)$
reduce from $O(n/\beta)$-GapSVP for $\gamma \geq O(n^{1/2})$	reduce from $O(n/\alpha)$-GapSVP for $\alpha \cdot q \geq O(n^{1/2})$
noise rate β	noise rate α
inverse period γ	$\alpha \cdot q$

Other results related to CLWE

Noise is necessary for hardness.

- The Lenstra-Lenstra-Lovász (LLL) algorithm can efficiently solve noiseless CLWE (or even CLWE with exponentially small noise [SongZadikBruna21]).
- Analogous to efficiently solving noiseless LWE with Gaussian elimination.
- Bypasses SQ-hardness since LLL inspects samples individually.

Subexponential algorithms for hCLWE with spacing $\gamma = o(n^{1/2})$.

- Simply compute covariance using $\exp(\gamma^2)$ many samples.
- Analogous to the Arora-Ge algorithm for LWE [AroraGe11].

Follow-up work

Hardness of learning “cosine neurons” [SZB21]

Observes that CLWE hardness also implies hardness of learning high-dimensional cosines (“cosine neurons”) of the form $f(\mathbf{x})=\cos(2\pi\gamma\langle\mathbf{w},\mathbf{x}\rangle)$ over the Gaussian input distribution if small (inverse-polynomial) **label noise** is added.

Previous work: Hardness of learning cosine neurons by SQ/gradient-based algorithms. [SongVempalaWilmesXie17,Shamir18]

Cosine neurons can be approximated (in the ℓ_2-norm) by poly-width **one-hidden-layer ReLU networks** if the input distribution is Gaussian. Hence, seemingly simple (NN-realizable) supervised learning tasks can be hard against a restricted class of algorithms. [SZB21] shows that the hardness applies to **any** polynomial-time algorithm.

Together with our result on hardness of learning Gaussian mixtures, this shows the versatility of CLWE/hCLWE as a primitive for showing hardness of improper learning.

Thank you!